Canon Research's Proposed Factorization Structure for Half-Sample Symmetric Filter Banks

Christopher M. Brislawn and Brendt E. Wohlberg
Los Alamos National Laboratory
Los Alamos, NM 87545

12 December 2000

1 Introduction

In a recently distributed report [1], E. Majani of Canon Research—France presented a mathematical structure and factoring algorithm for lifted factorization of half-sample symmetric (HS) perfect reconstruction filter banks. We comment on the proposed factorization, point out some technical errors, and propose our own solutions to these problems.

2 Inversion

As Margaret Lepley pointed out in her email to Majani on 29 Nov. 2000 (appended), aside from a number of notational inconsistencies, the block diagram in Figure 2 of [1], "Inverse Lifting Implementation," is *not* the mathematical inverse of the transformation depicted in Figure 1, "Forward Lifting Implementation." For instance, the last lifting step in Figure 2,

$$y_{2n+1} = y_{2n+1} - R(y_{2n+1-m_0}) \quad , \tag{1}$$

is *not* the inverse of the first lifting step in Figure 1, which in fact is identical to equation (1). For these two steps to be inverses of one another, one of them must be changed so as to *add* the rounded value; i.e.,

$$y_{2n+1} = y_{2n+1} + R(y_{2n+1-m_0}) \quad . \tag{2}$$

Convention (and consistency with the rest of Figure 1) would suggest changing the first step in Figure 1 to have the form of equation (2), but it isn't important for invertibility so long as one diagram adds $R(y_{2n+1-m_0})$ to y_{2n+1} and the other diagram subtracts the same quantity from y_{2n+1}.

Similarly, both lifting steps in the loop incremented by the parameter j in Figures 1 and 2 add the same quantity in both the forward and inverse lifting implementations. In particular, the lowpass step in the j loop is defined to be

$$y_{2n} = y_{2n} + R(\alpha_{0,j} \cdot y_{2n+m_j}) \tag{3}$$

in both Figures 1 and 2, whereas the rounded quantity should be added to y_{2n} in one diagram and subtracted in the other. Likewise, the highpass step in the j loop is

$$y_{2n+1} = y_{2n+1} + R(\beta_{0,j} \cdot (y_{2n} - y_{2n+2}) \tag{4}$$

in both Figures 1 and 2, whereas the rounded quantity should be added to y_{2n} in one diagram and subtracted in the other.

Finally, as Margaret pointed out, the first lifting step in Figure 2 is *always* a lowpass WSA lifting step whereas the last step in Figure 1 can be either a lowpass or a highpass WSA step. This implies that Figure 2 cannot be the inverse of Figure 1 in those cases where Figure 1 terminates with a highpass WSA step.

3 Reversibility

Document [1] presents the proposed lifting algorithm as being applicable in an implementation using symmetric signal extension [2, 3]. The nonlinear rounding operation, $R(x)$, needed for reversible implementation is not specified with any great precision; indeed, in Section 2.1.1 we read that

> The function $R(x)$ is generally any approximation of the variable x. $R(x)$ can be a rounding operator which rounds a real value x to an integer (such as the nearest integer), or it can be the identity: $R(x) = x$.

In fact, this is not true, and an integer-to-integer filter bank implementation using symmetric signal extension may not be reversible if the rounding operation is not chosen carefully. The reason this occurs is that the highpass-filtered subbands in a symmetric extension implementation are antisymmetric, and this property may not be preserved by some choices of rounding operation in lifting steps. This phenomenon creates situations in which the antisymmetric extension rules for highpass subband synthesis fail to reproduce exactly the (non-antisymmetric) subband that was generated by the nonlinear analysis filter bank.

For instance, Part 1 of the JPEG-2000 standard [4] uses the floor (i.e., greatest-integer) function for rounding, and the floor function does not preserve antisymmetry:

$$\lfloor -x \rfloor \neq -\lfloor x \rfloor \quad . \tag{5}$$

Nonetheless, the floor function is presented in Section 4.4 of [1] as part of an example of a preferred implementation.

Our proposed solution to this problem is to impose the restriction that rounding operations used in lifting implementations of reversible HS filter banks must preserve antisymmetry; i.e., they must satisfy

$$R(-x) = -R(x) \quad . \tag{6}$$

It follows that any HS integer-to-integer filter bank lifted from the Haar filter bank that uses a rounding operation satisfying (6) will be reversible in implementations utilizing symmetric extension at signal boundaries. (Strictly speaking, equation (6) need not be satisfied in the lowpass lifting steps, so one could in principle use different rounding rules in the lowpass and highpass lifting steps, if there were motivation for doing so.)

One example of such a function is fractional part truncation, as exemplified by the C programming language's rule for casting floating point types to integer types [5]:

$$R(x) = (\texttt{int})x \quad . \tag{7}$$

Fractional part truncation satisfies equation (6).

Finally, if an additive offset is included in the rounding operation, it must be alternately added or subtracted, depending on the sign of x, in such a way as to preserve equation (6). Given a rounding operation, $R'()$, that satisfies equation (6), we can supplement it with an additive offset, β, in ways that will preserve equation (6). For instance, define $R()$ as:

$$R(x) = \begin{cases} R'(x+\beta) & , \quad x \geq 0 \\ R'(x-\beta) & , \quad x < 0 \quad . \end{cases} \tag{8}$$

It can be verified that $R()$ as defined in equation (8) satisfies equation (6).

References

[1] E. Majani, "Lifting implementation of even-length symmetric filters," Tech. Rep. ISO/IEC JTC1/SC29/WG1N1914, Int'l. Org. for Standardization, 9 Nov. 2000. Posted to WG1 web site on 22 Nov. 2000.

[2] C. M. Brislawn, "Classification of nonexpansive symmetric extension transforms for multirate filter banks," *Appl. Comput. Harmonic Anal.*, vol. 3, pp. 337–357, 1996.

[3] ISO/IEC JTC1/SC29/WG1, *JPEG-2000 Image Coding System, Part 2*, ISO/IEC Standard 15444-2 (Committee Draft), Int'l. Org. for Standardization, Dec. 2000.

[4] ISO/IEC JTC1/SC29/WG1, *JPEG-2000 Image Coding System, Part 1*, ISO/IEC Standard 15444-1 (Final Draft Int'l. Standard), ITU-T Rec. T.800, Int'l. Org. for Standardization, 2000.

[5] B. W. Kernighan and D. M. Ritchie, *The C Programming Language*. Englewood Cliffs, NJ: Prentice Hall, 1978.

Subject: [Fwd: Lifting Implementation of HSS/HSA]
Date: Wed, 06 Dec 2000 03:52:34 -0500
From: Margaret Lepley <mlepley@mitre.org>
Organization: The MITRE Corporation
To: C M Brislawn <brislawn@lanl.gov>

Chris --

Here's the email I sent Eric on his writeup. After reading further I unconfused myself and could better distinguish the correct part from the errors. But this points to where the major inconsistencies in the original document.

Margaret

-------- Original Message --------
Subject: Lifting Implementation of HSS/HSA
Date: Wed, 29 Nov 2000 22:15:24 -0500
From: Margaret Lepley <mlepley@mitre.org>
Organization: The MITRE Corporation
To: 'MAJANI Eric' <majani@crf.canon.fr>

Eric --

I'm finally getting a chance to read your paper. So far I really like the organization and explanation, but section 4.1.5 is causing me problems.

You say you want k_max-1 lifting steps total. (Not pairs of lifting steps as in sect 4.1.3.) That seems fine, but then you use the notation 2k-1 and 2k in EQ 22&23 ... which seems to indicate that they are indeed in pairs and that a total of 2(k_max-1) liftings steps will be performed.

So I looked at Figure 1, which indicates k_max-1 total steps. But I hope you notice that first WSA lifting step uses alpha_1,j, then the next one uses alpha_4,j, and then alpha_5,j, and then alpha_8,j, etc. Weird. So alpha_2,j, alpha_3,j, alpha_6,j, alpha_7,j etc don't ever need to be defined. Is this what you wanted? Or maybe that really is standard in the literature (which I am unfortunately not familiar with). My personal choice would have been to use just k to index the equations.

I also notice that since you start with a lowpass WSA step and just completed a lowpass step to end the previous segment, that gives two lowpasses in a row. I don't know enough about it to comment, but it did make me wonder. Especially in combination with what I noticed when inspecting the inverse transform flowchart.

Figure 2 is definitely not the inverse of Figure 1 :-(. Since the forward transform can end on either a lowpass or highpass step depending upon whether k_max is even or odd ... that means the inverse transform must start with a lowpass or highpass depending upon k_max. But Figure 2 always start with a lowpass step. Moreover, as I mentioned in the last paragraph, the first forward WSA step is lowpass ... so the last inverse step in that section must be lowpass.

And I think the m_j initialization needs to be -1 rather than 1. m_LO=1, I think (?) so if j=LO-1, m_j=-1.

I'm sorry it's so late in the game, but since what you have already is so nicely laid out, I thought it would be worth fixing.

Look forward to seeing you in New Orleans,
Margaret

Margaret A. Lepley	email: mlepley@mitre.org
The MITRE Corporation, Mailstop K211	voice: 781-271-2770
202 Burlington Road	fax: 781-271-2721
Bedford, MA 01730	